In [31]:

```
%matplotlib inline
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import mpl_toolkits
```

In [32]:

```
data = pd.read_csv("C:\\Temp\\house_data.csv")
```

In [33]:

```
data.head()
```

Out[33]:

	id	date	price	bedrooms	bathrooms	sqft_living	sqft_lot	floors
0	7129300520	20141013T000000	221900.0	3	1.00	1180	5650	1.0
1	6414100192	20141209T000000	538000.0	3	2.25	2570	7242	2.0
2	5631500400	20150225T000000	180000.0	2	1.00	770	10000	1.0
3	2487200875	20141209T000000	604000.0	4	3.00	1960	5000	1.0
4	1954400510	20150218T000000	510000.0	3	2.00	1680	8080	1.0

5 rows × 21 columns

In [34]:

```
data.describe()
```

Out[34]:

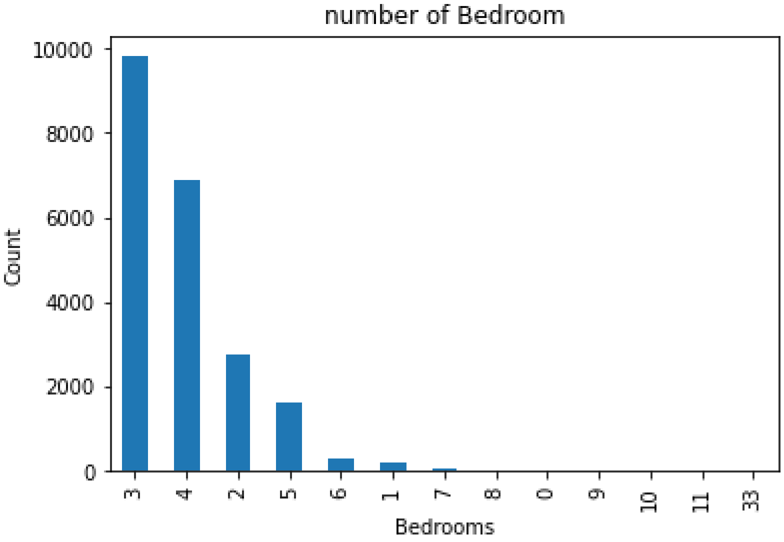
	id	price	bedrooms	bathrooms	sqft_living	sqft_lot
count	2.161300e+04	2.161300e+04	21613.000000	21613.000000	21613.000000	2.161300e+04
mean	4.580302e+09	5.400881e+05	3.370842	2.114757	2079.899736	1.510697e+04
std	2.876566e+09	3.671272e+05	0.930062	0.770163	918.440897	4.142051e+04
min	1.000102e+06	7.500000e+04	0.000000	0.000000	290.000000	5.200000e+02
25%	2.123049e+09	3.219500e+05	3.000000	1.750000	1427.000000	5.040000e+03
50%	3.904930e+09	4.500000e+05	3.000000	2.250000	1910.000000	7.618000e+03
75%	7.308900e+09	6.450000e+05	4.000000	2.500000	2550.000000	1.068800e+04
max	9.900000e+09	7.700000e+06	33.000000	8.000000	13540.000000	1.651359e+06

In [35]:

```
data['bedrooms'].value_counts().plot(kind='bar')
plt.title('number of Bedroom')
plt.xlabel('Bedrooms')
plt.ylabel('Count')
sns.despine
```

Out[35]:

```
<function seaborn.utils.despine(fig=None, ax=None, top=True, right=True, left
=False, bottom=False, offset=None, trim=False)>
```

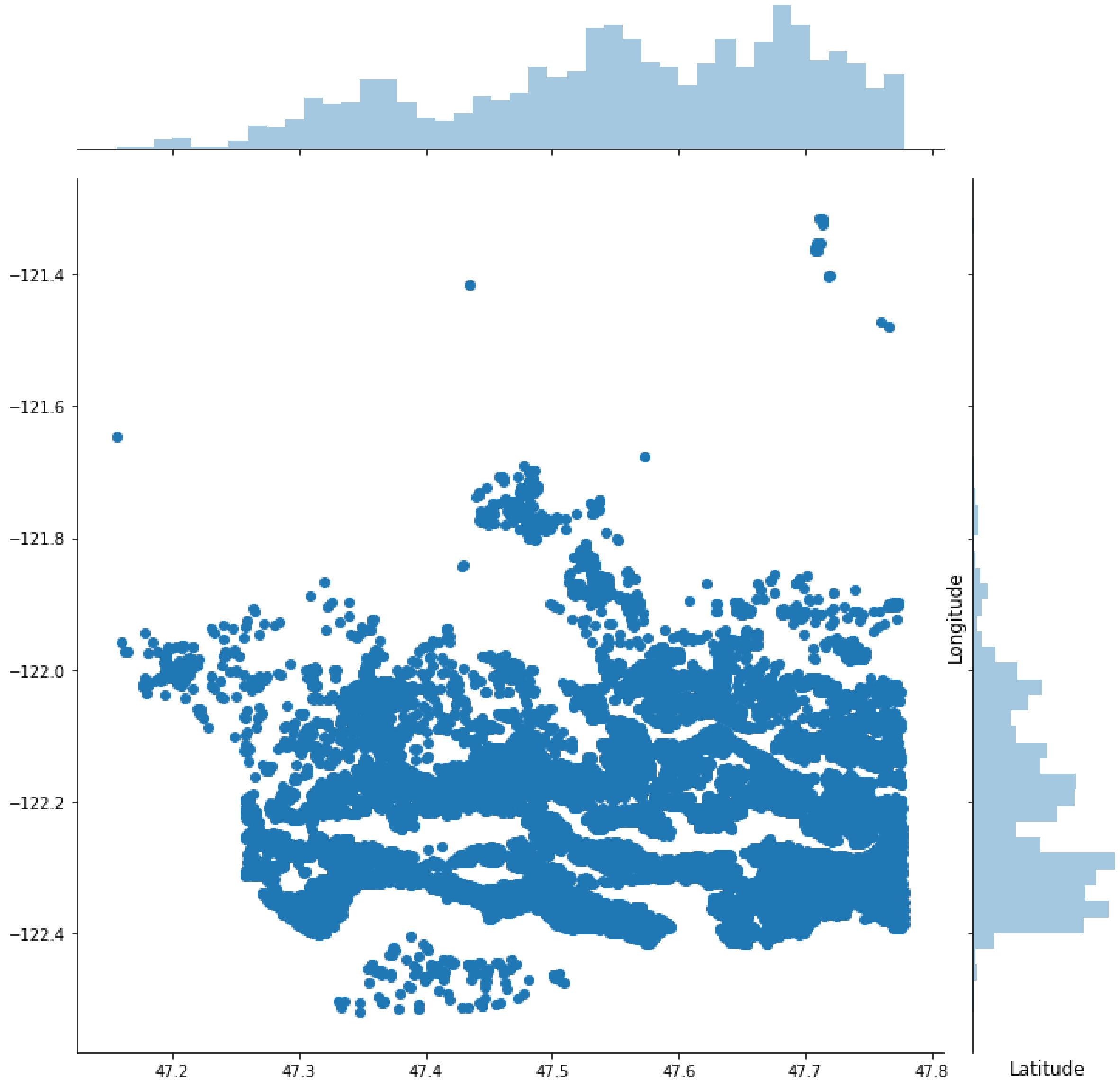


In [36]:

```
plt.figure(figsize=(10,10))
sns.jointplot(x=data.lat.values, y=data.long.values, size=10)
plt.ylabel('Longitude', fontsize=12)
plt.xlabel('Latitude', fontsize=12)
plt.show()
plt1 = plt()
sns.despine
```

```
C:\Users\ddeha\anaconda3\lib\site-packages\seaborn\axisgrid.py:2264: UserWarn
ing: The `size` parameter has been renamed to `height`; please update your co
de.
  warnings.warn(msg, UserWarning)
```

```
<Figure size 720x720 with 0 Axes>
```

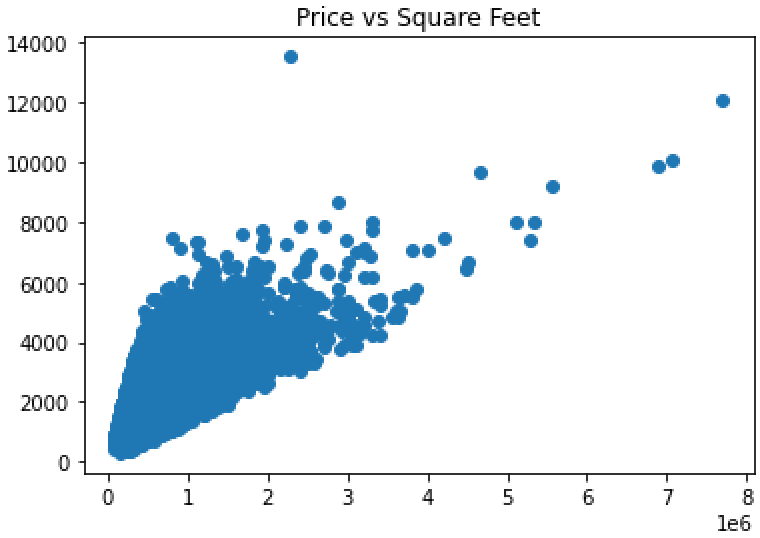


```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-36-67b21dd0040b> in <module>
      4 plt.xlabel('Latitude', fontsize=12)
      5 plt.show()
----> 6 plt1 = plt()
      7 sns.despine

TypeError: 'module' object is not callable
```

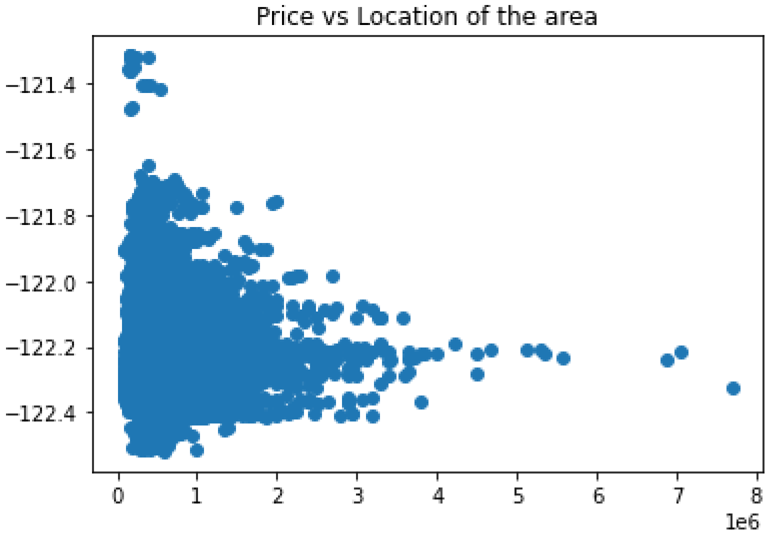
In [37]:
```
plt.scatter(data.price,data.sqft_living)
plt.title("Price vs Square Feet")
```

Out[37]:
```
Text(0.5, 1.0, 'Price vs Square Feet')
```



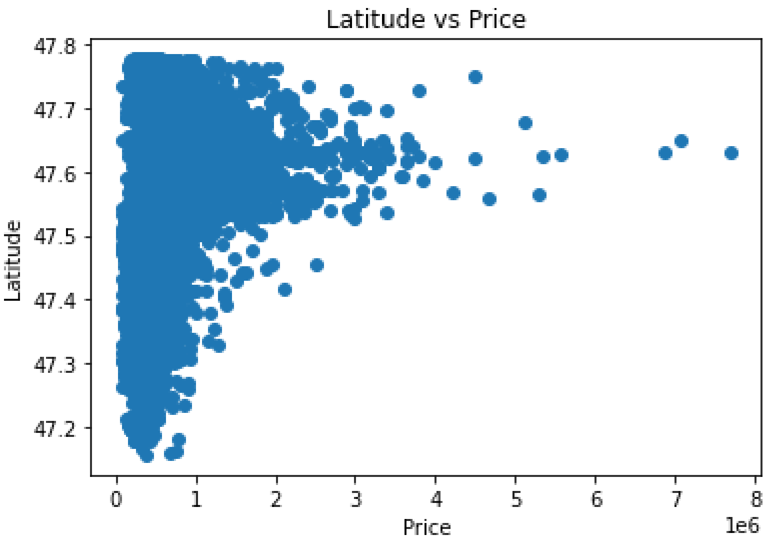
In [38]:
```
plt.scatter(data.price,data.long)
plt.title("Price vs Location of the area")
```

Out[38]:
```
Text(0.5, 1.0, 'Price vs Location of the area')
```

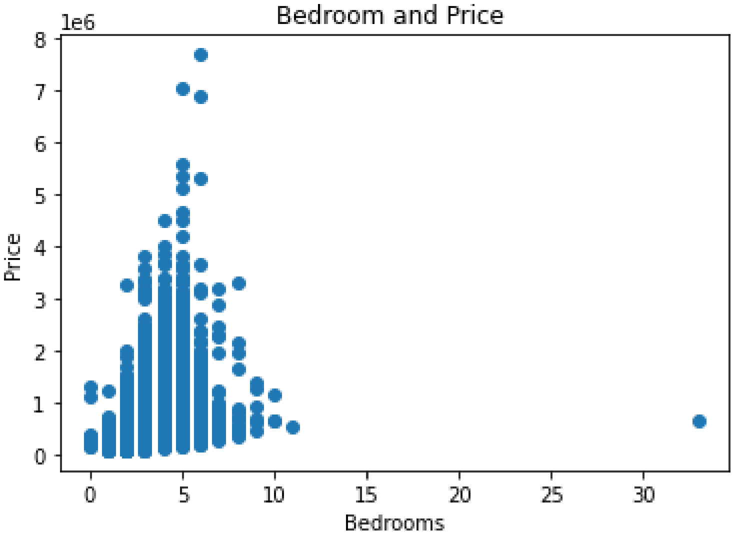


In [39]:
```
plt.scatter(data.price,data.lat)
plt.xlabel("Price")
plt.ylabel('Latitude')
plt.title("Latitude vs Price")
```

Out[39]:
```
Text(0.5, 1.0, 'Latitude vs Price')
```



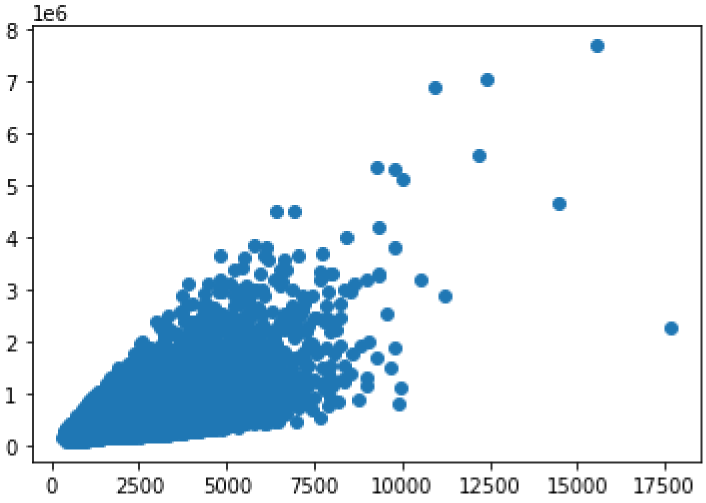
In [40]:
```
plt.scatter(data.bedrooms,data.price)
plt.title("Bedroom and Price ")
plt.xlabel("Bedrooms")
plt.ylabel("Price")
plt.show()
sns.despine
```



Out[40]:
```
<function seaborn.utils.despine(fig=None, ax=None, top=True, right=True, left
=False, bottom=False, offset=None, trim=False)>
```

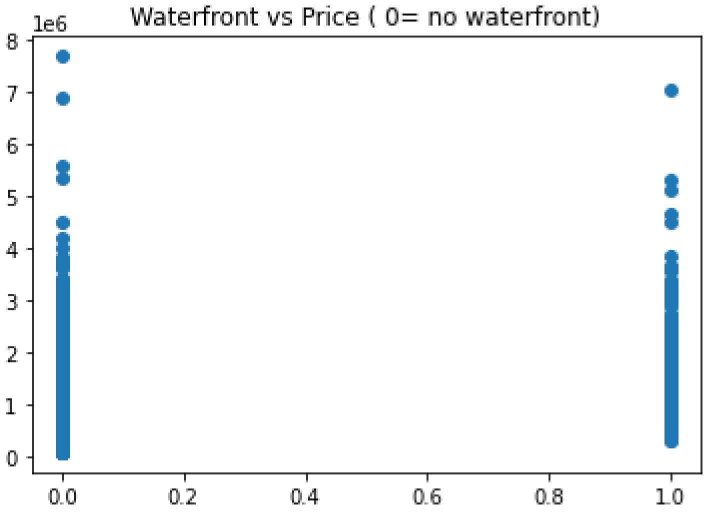
```
In [41]: plt.scatter((data['sqft_living']+data['sqft_basement']),data['price'])
```

```
Out[41]: <matplotlib.collections.PathCollection at 0x15ee3af6e80>
```



```
In [42]: plt.scatter(data.waterfront,data.price)
         plt.title("Waterfront vs Price ( 0= no waterfront)")
```

```
Out[42]: Text(0.5, 1.0, 'Waterfront vs Price ( 0= no waterfront)')
```



```
In [43]: train1 = data.drop(['id', 'price'],axis=1)
```

In [44]:

```
train1.head()
```

Out[44]:

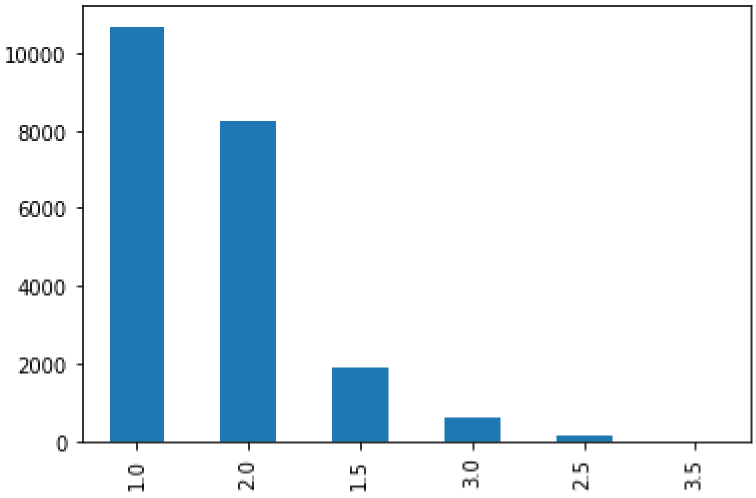
	date	bedrooms	bathrooms	sqft_living	sqft_lot	floors	waterfront	view	condit
0	20141013T000000	3	1.00	1180	5650	1.0	0	0	
1	20141209T000000	3	2.25	2570	7242	2.0	0	0	
2	20150225T000000	2	1.00	770	10000	1.0	0	0	
3	20141209T000000	4	3.00	1960	5000	1.0	0	0	
4	20150218T000000	3	2.00	1680	8080	1.0	0	0	

In [45]:

```
data.floors.value_counts().plot(kind='bar')
```

Out[45]:

```
<AxesSubplot:>
```

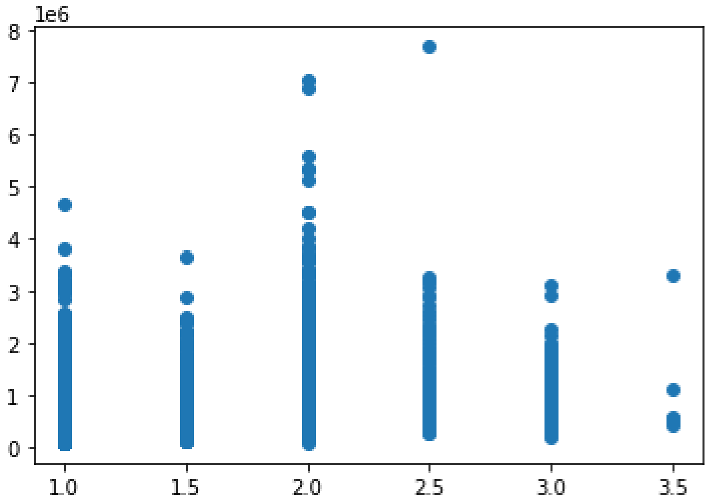


In [46]:

```
plt.scatter(data.floors,data.price)
```

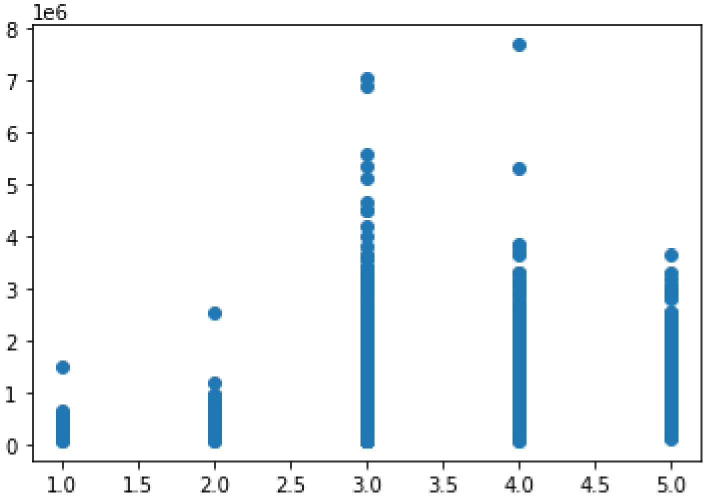
Out[46]:

```
<matplotlib.collections.PathCollection at 0x15ee3715a60>
```



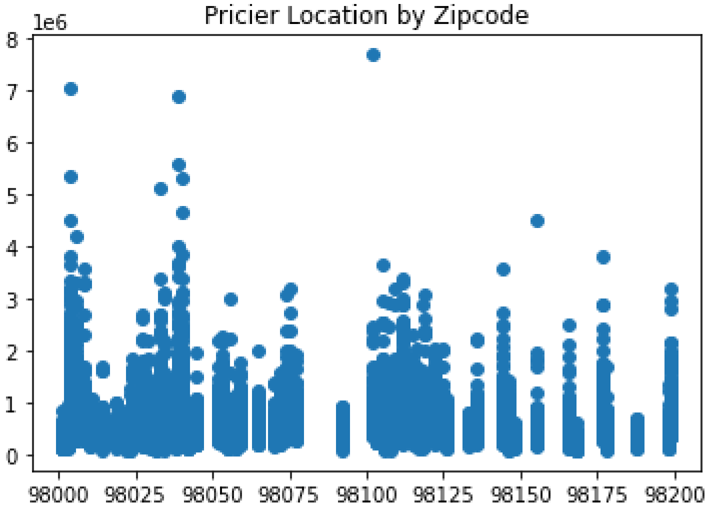
In [47]:
```
plt.scatter(data.condition,data.price)
```

Out[47]:
```
<matplotlib.collections.PathCollection at 0x15ee37791c0>
```



In [48]:
```
plt.scatter(data.zipcode,data.price)
plt.title("Pricier Location by Zipcode")
```

Out[48]:
```
Text(0.5, 1.0, 'Pricier Location by Zipcode')
```



In [49]:
```
from sklearn.linear_model import LinearRegression
```

In [50]:
```
reg = LinearRegression()
```

In [51]:
```
labels = data['price']
conv_dates = [1 if values == 2014 else 0 for values in data.date ]
data['date'] = conv_dates
train1 = data.drop(['id', 'price'],axis=1)
```

In [52]:
```
from sklearn.model_selection import train_test_split
```

In [53]:

```
x_train , x_test , y_train , y_test = train_test_split(train1 , labels , test_
size = 0.10,random_state =2)
```

In [54]:

```
reg.fit(x_train,y_train)
```

Out[54]:

```
LinearRegression()
```

In [55]:

```
reg.score(x_test,y_test)
```

Out[55]:

```
0.7320342760357544
```

In [56]:

```
from sklearn import ensemble
clf = ensemble.GradientBoostingRegressor(n_estimators = 400, max_depth = 5, mi
n_samples_split = 2,
          learning_rate = 0.1, loss = 'ls')
```

In [57]:

```
clf.fit(x_train, y_train)
```

Out[57]:

```
GradientBoostingRegressor(max_depth=5, n_estimators=400)
```

In [58]:

```
clf.score(x_test,y_test)
```

Out[58]:

```
0.9204035711689568
```